# Generating Collaborative Behaviour through Plan Recognition and Planning

# Generating Collaborative Behaviour through Plan Recognition and Planning

**Christopher Geib**
Department of Computer Science
Drexel University
Philadelphia, PA 19104, USA
cgeib@drexel.edu

**Bart Craenen**
School of Computing Science
Newcastle University
Newcastle NE1 7RU, England, UK
Bart.Craenen@newcastle.ac.uk

**Ronald P. A. Petrick**
Department of Computer Science
Heriot-Watt University
Edinburgh EH14 4AS, Scotland, UK
R.Petrick@hw.ac.uk

## Abstract

This paper presents a framework for integrated plan recognition and automated planning, to produce collaborative behaviour for one agent to help another agent. By observing an "initiator" agent performing a task, the plan recognizer hypothesises how a "supporter" agent could help the initiator by proposing a set of subgoals to be achieved. A lightweight negotiation process mediates between the two agents to produce a mutually agreeable set of goals for the supporter. The goals are passed to a planner which builds an appropriate sequence of actions for satisfying the goals. The approach is demonstrated in a series of experimental scenarios.

## Introduction

The ability of an agent to *help* another agent is a desirable attribute when designing artificial entities, such as robots, that must operate together with humans in real-world environments. Indeed, the idea of building assistive agents that must work alongside humans in a cooperative fashion has been a long-standing goal of artificial intelligence and robotics since its earliest days. However, the task of deciding when and how to help another agent can be difficult. Effective helping involves recognising the goals or intentions of other agents, reasoning about opportunities to contribute to existing plans, generating appropriate actions, and potentially communicating such information to the agents involved. In the worst case, identifying an opportunity to help, and generating an appropriate response, may require reasoning over the entire joint space of goals and actions for all the agents.

While the computational cost of reasoning about cooperative action in its most general form may be entirely impractical, constrained forms of reasoning do exist that could be used as the basis for helpful behaviour. For instance, consider the case of two agents setting a table for dinner, where the first agent sets the plates and glasses, and the second agent sets the knives, forks, and spoons. The subgoals pursued by each agent are disjoint but together they contribute to a shared overall goal. Moreover, each action is performed by a single agent, with no action requiring the joint coordination of multiple agents (e.g., two agents lifting a table). Finally, the order of subgoal achievement is independent of the actions of the other agent (e.g., it makes no difference if the knives are placed before the forks or vice versa).

In this paper we consider scenarios of the above form, where one agent, called the *supporter*, must decide how to act to help a second agent, called the *initiator*, achieve its goals. While the supporter is considered to be an artificial agent, no assumption is made about the initiator which may either be a human or artificial agent. In this work, we consider goals which can be decomposed as in the above example, and tasks that consist of independent sequences of actions for each agent. While such conditions may appear to be restrictive, they nevertheless characterise a useful collection of problem scenarios whose solution is far from trivial: the goals of the initiator must be identified and suitable subgoals must be appropriately selected for the supporter to achieve.

To do so, we combine *plan recognition* and *automated planning*, together with a lightweight negotiation process for ensuring that a set of supporter goals is acceptable to both agents. Plan recognition and plan generation are provided by two existing frameworks: the ELEXIR plan recognizer (Geib 2009) and the PKS planner (Petrick and Bacchus 2002; 2004). As such, we focus on the high-level (symbolic) reasoning involved in this task, rather than the low-level processes (e.g., involving continuous models or geometric reasoning) that are part of the design of certain artificial agents like robots. Moreover, the novelty of the approach arises from the particular combination of these two general techniques (i.e., plan recognition and planning), rather than the specific tools used to implement them.

In this approach, the supporter will infer the high-level plans of the initiator and identify possible subgoals that contribute to the initiator's plan. Pairs consisting of the initiator's hypothesised high-level goal, and a candidate subgoal, will then be proposed to the initiator as possible helpful subgoals that the supporter could accomplish. This involves a directed search that first attempts to find the hypothesised goal of the initiator, followed by a search of the remaining subgoals that could be performed by the supporter.

Once negotiation is complete, the agreed upon goals are passed to an automated planner which constructs an independent sequence of actions for the supporter to execute to help the initiator. In particular, no centralised planning or scheduling component is used to enforce collaborative behaviour through joint plans. For example, after observing the initiator place spoons on the table, the supporter might infer that the initiator is setting the table, and that the plates

still need to be set. After confirming with the initiator that setting the plates would help the initiator achieve its goals, the supporter can build and execute a plan for this subgoal. However, if the initiator denies either the hypothesised goal (e.g., the initiator is instead placing the spoons onto the table in order to polish them) or the proposed subgoal (e.g., only bowls need to be set), then an alternative goal/subgoal pair could be proposed to find another way to help the initiator.

The rest of this paper is organised as follows. First, we review the relevant related work. Next, we highlight the main components in our approach, notably the ELEXIR plan recognizer and the PKS planner, and discuss the integration of these systems. We then present the results of our approach tested in three experimental domains. Finally, we discuss the limitations of our approach and highlight future directions.

## Related Work

The idea of constructing cooperative agents has been a longstanding area of research (Nwana 1996). Moreover, the task of building artificial agents (especially robots) that can proactively achieve goals has been an active area of study (Schrempf et al. 2005; Pandey, Ali, and Alami 2013), as has the idea of human-robot collaboration (Bauer, Wollherr, and Buss 2008; Chandrasekaran and Conrad 2015). As a result, this work follows a long tradition of prior approaches addressing various aspects of this complex problem.

The general idea of agents that help other agents (including humans) has variously been viewed as a primary property of a plan, or as implicit in multiagent actions. For example, (Pollack 1990; Lochbaum, Grosz, and Sidner 1990) explicitly reason about coordination and helping in the form of shared plans and mutual beliefs. However, establishing agreement of such plans or beliefs has typically relied on shared knowledge which has long been a stumbling block of such theories. Similarly, action representations for multiagent joint actions (i.e., actions that require two or more agents for their execution) (Brafman and Domshlak 2008; Boutilier and Brafman 2001) could be used to model situations where one agent helps another agent. However, these representations do not address the case where helping is not a consequence of such multiagent joint actions.

There has also been significant prior research on a variety of approaches to multiagent planning (e.g., (Nau 2007; Brenner 2003; Brafman and Domshlak 2008; Crosby, Jonsson, and Rovatsos 2014)), along with work on the decentralised solving of constraint optimisation problems (Modi et al. 2003). Approaches have also considered the use of plan recognition (Talamadupula et al. 2014) and intent recognition (Karpas et al. 2015) as a means of coordinating human-robot teams. The idea of ambient intelligence (Augusto 2007) also has connections to the problem of designing systems that proactively aid humans in achieving their goals.

The role of natural language dialogue as an effective means of coordinating actions between a robot and a human has also been previously studied (Fong, Thorpe, and Baur 2003). Moreover, the combination of natural language and goal inference has been explored for the task of selecting actions to contribute to an ongoing task, or for correcting the action of a human already engaged in the task (Foster et al. 2008; Giuliani et al. 2010). Finally, hybrid architectures have been used to integrate diverse components with different representational requirements, particularly when a robot must cooperate with a human (Hawes et al. 2007; Kennedy et al. 2007; Zender et al. 2007).

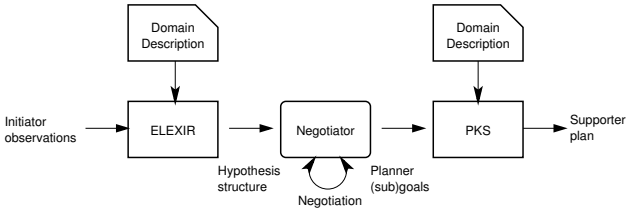


Figure 1: Components and interactions in the framework.

## A Framework for Collaborative Behaviour

We now present our approach to collaborative behaviour by describing the main components in our work: the Engine for LEXicalized Intent Recognition (ELEXIR) plan recognizer, the negotiation process, and the Planning with Knowledge and Sensing (PKS) planner. The relationship between these components is shown in Figure 1 and discussed below.

### Plan Recognition with ELEXIR

We begin by first distinguishing between work in *activity recognition* (also called *goal recognition* (Liao, Fox, and Kautz 2005; Hoogs and Perera 2008; Blaylock and Allen 2003)) and *plan recognition* in this context. Activity recognition is the creation of a single unstructured label that represents the overarching goal of the activity being observed. For example, such an algorithm would recognize a sequence of pick and place actions of forks, knives, spoons, and plates as an instance of setting the table. This kind of single label is insufficient for our purposes. We need to know the steps in the plan already completed by the initiator, and whether or which future subgoals the supporter can still contribute to.

In contrast, plan recognition attempts to identify not only the goal being pursued by the agent but also the subgoals of the plan that have already been accomplished, and those that are anticipated to be part of the plan in the future. Thus, a plan recognition algorithm is able to produce the complete unexecuted frontier of a hierarchical plan (Kautz 1991; Blaylock and Allen 2003; Geib 2009). For example, following observations of picking and placing forks followed by knives, such a system could identify that the goal was to set the table, the current subgoal was to set the knives, and that in the future, the agent would be setting spoons and plates. These predicted future subgoals are required to effectively reason about possible collaborative contexts.

In this work, we use ELEXIR (Geib 2009) to perform the kind of plan recognition described above. ELEXIR is a probabilistic plan recognition system that views the problem as an instance of parsing a probabilistic grammar. As such, ELEXIR takes as input a formal probabilistic grammar that specifies the set of plans to be recognized and a set of observed actions. ELEXIR represents its plans using Combina-

**set-forks** :=*SetTable*/{*SetKnives*, *SetSpoons*, *SetPlates*, *SetGlasses*} |
(*CleanForks*/{*PutAwayForks*})/{*WashForks*}.
**set-knives** :=*SetKnives*. **set-spoons** := *SetSpoons*.
**set-plates** :=*SetPlates*. **store-forks** := *PutAwayForks*.

Figure 2: Portion of a CCG action grammar in ELEXIR.

tory Categorial Grammars (CCGs) (Steedman 2000). While a full discussion of CCGs in ELEXIR is not possible in the space available, we include an example to aid our discussion.

Figure 2 shows a portion of a CCG action grammar that captures a plan for *SetTable* and *CleanForks*. *SetTable*, *SetKnives*, *SetSpoons*, *SetPlates*, *SetGlasses*, *CleanForks*, *PutAwayForks*, and *WashForks* are all symbols that represent goals or subgoals within the plan library. As such, this grammar already encodes some abstraction in the plans. For example, as we will see, a **set-forks** action can be realized by the planner as a sequence of four lower level actions. We assume activity recognition is able to produce observations of the defined high-level actions (e.g., **set-forks**, **set-knives**, **set-spoons**,.... ) from observations of lower level actions. We could have encoded the grammar at a finer granularity, but this would have added significant unnecessary complexity to the example. Further, because the actions are being executed by the initiator, this would not have eliminated the need for assuming activity recognition of the observed actions. Thus, our example grammar is presented at this abstract level.

That said, the grammar doesn't make commitments about the level of subgoal abstraction. For example, *WashForks* is likely a complex subplan in its own right. Finally, we note that while ELEXIR does support actions with variable arguments, all of our examples use propositional actions, again to simplify the discussion (see (Geib 2009) for more details about ELEXIR's handling of non-propositional actions).

The grammar in Figure 2 specifies that two possible plans can account for an observation of the action **set-forks**: *SetTable* and *CleanForks*. *SetTable* requires that *SetKnives*, *SetSpoons*, *SetPlates*, and *SetGlasses* follow the observed occurrence of **set-forks**, but the subgoals are unordered with respect to each other. *CleanForks* can explain the observed **set-forks**, but only if *WashForks* follows it, and followed by *PutAwayForks*.

Given a set of observed actions, and a formal grammar as above, ELEXIR produces the complete set of hierarchical plan structures that conforms to the grammar and is consistent with the observations, along with a probability for each. These structures represent the hypothesised plans being executed by the agent. Using it we can extract an ordered set of subgoals from each hypothesis that must still be executed for the goal to be achieved, associated with the probability of the hypothesis.

Note that ELEXIR supports both the possibility that a given agent can be pursuing multiple plans as well as the possibility of partially ordered plans. Therefore, for this discussion we will represent a hypothesis produced by ELEXIR as a tuple of the form:

$$(P, [\{G_i : \{\text{sg}_1, ..., \text{sg}_n\}^*\}^+]),$$

where $P$ is the probability of the hypothesis, $G_i$ is the goal of the hypothesised plan, and $\text{sg}_j$ the remaining sets of possibly partially ordered subgoals that must be achieved for $G_i$ to be completed. The $\text{sg}_j$ within one set of braces are treated as unordered with respect to each other, but all the $\text{sg}_j$ within one set must be achieved before those in the next set.

Thus, the three hypotheses from the table setting example (after observing the setting of forks) might be captured as:

$(.95, [\{\textit{SetTable} : \{\textit{SetKnives}, \textit{SetSpoons}, \textit{SetPlates}, \textit{SetGlasses}\}\}]),$
$(.045, [\{\textit{CleanForks} : \{\textit{WashForks}\}\{\textit{PutAwayForks}\}\}]),$
$(.005, [\{\textit{CountingForks} : \{\}\}]).$

The first tuple captures the hypothesis that with 95% probability the agent is following a plan to set the table, and still has the subgoals to set the knives, spoons, and plates. These subgoals are unordered with respect to each other within the plan. The second tuple captures the hypothesis that with 4.5% probability the agent is cleaning the forks and still needs to wash them and put them away, in that order. The third tuple captures the hypothesis that with only 0.5% probability the agent is simply counting the forks and is done with its plan. Thus, each hypothesis provides us with access to the probability of the plans being executed, the goals they are intended to achieve, and the subgoals in the plan that have yet to be achieved. This is precisely the information that we need in order to identify opportunities where the supporter can help the initiator. We discuss how this is done in the next section.

## Subgoal Identification and Negotiation

In order to efficiently negotiate collaboration, a supporter must first confirm that it understands the objective of the initiator's high-level plan. Without this confirmation, the supporter might waste significant amounts of time suggesting subgoals that it could achieve, but that may not contribute to the initiator's goal and plan. Using the hypothesis structures from ELEXIR this can be done in a straightforward way.

In the case where a single plan is being pursued by the initiator, sorting the hypotheses by their probabilities ranks the goals of the plan being pursued. This makes it relatively easy for the supporter to verify the initiator's actual plan by a simple query to the initiator.

Having thus identified the goal of the initiator's plan, the supporter can then attempt to identify a future subgoal within those hypotheses that share the identified goal. Returning to our example, when considering the hypotheses for the setting of forks, the first hypothesis is the most likely:

$(.95, [\{\textit{SetTable} : \{\textit{SetKnives}, \textit{SetSpoons}, \textit{SetPlates}, \textit{SetGlasses}\}\}]).$

If the initiator confirms that SetTable is in fact the goal of its plan, the supporter could then suggest that it take on the subgoals of *SetKnives*, *SetSpoons*, *SetPlates*, and *SetGlasses*, or some subset thereof. As we will see in Section , a maximally helpful agent would volunteer to do all of these subgoals. Note, however, that the negotiation process could also result in a number of other outcomes, whereby the supporter agrees to some subset of the subgoals, or none of them at all.

In effect, the process of negotiating collaboration between the initiator and the supporter is then a directed search: first to identify the goal of the initiator's plan, and then to find appropriate subgoals from the set of known unaccomplished subgoals of the plan the supporter has inferred for the goal.

### Automated Planning with PKS

Once negotiation is complete and has produced a set of subgoals for helping the initiator, the supporter must generate a concrete sequence of actions to execute in the world. To do so, we use the off-the-shelf PKS planning system.

PKS (Planning with Knowledge and Sensing) (Petrick and Bacchus 2002; 2004) is a contingent planner that builds plans using incomplete information and sensing. PKS operates at the knowledge level (Newell 1982) by reasoning about how the planner's knowledge state changes due to action. PKS is based on a generalisation of STRIPS (Fikes and Nilsson 1971). In PKS, the planner's knowledge state (rather than the world state) is represented by a set of databases, each of which models a particular type of knowledge. The contents of each database have a formal interpretation in a modal logic of knowledge. Actions can modify the databases, which has the effect of updating the planner's knowledge. To ensure efficient inference, PKS restricts the type of knowledge (especially disjunctions) it can represent. The information in PKS's databases can also be incomplete, and PKS does not make a closed world assumption. PKS also supports features like functions and run-time variables that arise in real-world planning scenarios.

Like other planners, a PKS planning domain consists of an initial state, a set of actions, and a set of goals. The initial state is simply the planner's initial knowledge (databases). Goals specify the knowledge conditions that the planner is trying to achieve, formed from the supporter's agreed upon subgoals through a syntactic compilation process which transforms the subgoals into a form understandable by PKS. Actions in PKS are modelled by their preconditions that query the planner's knowledge state, and effects that change the knowledge state by updating particular databases. Plans are constructed by a forward-chaining heuristic search, starting from the initial knowledge state, and continuing until the goal conditions are satisfied or the search fails.

For instance, Figure 3 shows two PKS actions taken from our experimental domains (Section ). A precondition `K`($\phi$) queries PKS's knowledge to determine if the planner knows $\phi$, while an effect that references `Kf` updates PKS's database of known world facts. Using these actions, a plan such as:

```
grasp(left,drawer,fork1),
putdown(left,table_pos1,fork1),
grasp(left,drawer,fork2),
putdown(left,table_pos2,fork2)
```

might be built in support of a goal to put forks on the table.

### Integration and Operation

From a technical point of view, both ELEXIR and PKS are implemented as C++ libraries, with ELEXIR structured into core and recognizer parts. Both libraries expose a user interface through ZeroC's Internet Communication Engine (ICE), a modern distributed computing platform (Henning 2004). This allows both ELEXIR and PKS to be used as standalone servers by a client application implementing the framework, in a traditional client-server architecture.

```
action grasp(?h : hand, ?l : loc, ?o : obj)
   preconds: K(graspable(?o, ?h)) &
             K(objectAt(?o, ?l)) &
             K(holding(?h) = nil)
   effects:  add(Kf, holding(?h) = ?o),
             del(Kf, objectAt(?o, ?l))

action putdown(?h : hand, ?l : loc, ?o : obj)
   preconds: K(holding(?h) = ?o)
   effects:  add(Kf, objectAt(?o, ?l)),
             add(Kf, holding(?h) = nil)
```

Figure 3: PKS actions in the experimental domain.

Figure 1 illustrates the flow of control between the plan recognition, negotiation, and automated planning components in the framework. At system initialisation time, both the plan recognizer and planner are provided with domain-dependent knowledge in the form of their respective domain descriptions. This information is minimally aligned to ensure interoperation between these components (see below).

The process then starts with the supporter observing actions performed by the initiator. These observations are fed into ELEXIR, which produces a set of hypotheses about the initiator's high-level plan, as goal/subgoal pairs, in a hypothesis structure. This structure is then handed over to the negotiation process which mediates the negotiation between the supporter and initiator. Negotiation proceeds by applying directed search to the hypothesis structure to produce a set of goals for the planner. In the final step, PKS uses these goals to attempt to generate a plan to be executed by the supporter.

## Experimental Demonstration and Validation

We now present three scenarios, based on the table setting running example, as an experimental demonstration of the proposed framework, integrated as depicted in Figure 1. The underlying domain setting for the three scenarios remains the same: an initiator agent has begun setting a table for a dinner for two people, where each place setting should include a knife, fork, spoon, plate, and glass. The aim of the supporter agent is to help the initiator complete the overall goal of setting the table. Knowledge about the operating environment and the requirements for setting tables is supplied to both the plan recognizer and the planner using appropriate domain descriptions, as detailed earlier.

The observations provided to the ELEXIR plan recognizer remain the same for each scenario: one by one the initiator picks up two forks and two knives and puts them down in their appropriate positions on the table. The scenarios differ in the way these observations are interpreted, and the, way in which differences in the negotiation process can lead to the identification of different subgoals, and how that can affect the resulting plans. The process ends when the planner builds a plan for the supporter to perform, based on the goals identified during the negotiation process.

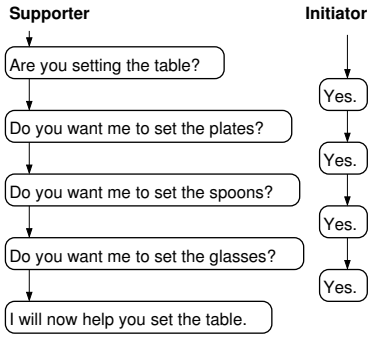


Figure 4: Negotiation in Scenario 1.

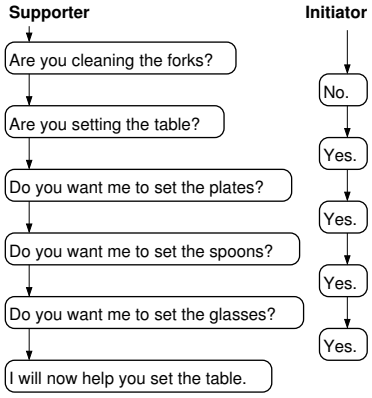


Figure 5: Negotiation in Scenario 2.

For each scenario the correctness of the approach is validated during experimentation. Validation focuses on two points in the process: first, whether the plan recognizer interprets the observation correctly, and, second, whether the planner produces the correct plans. Validation of this form is possible in this case because the example scenarios are designed so that we know, beforehand, what the negotiation process should look like and, as a consequence, how the supporter is supposed to help the initiator set the table.

The computational requirements for all three example scenarios are minimal. Both plan recognition and planning in this domain context takes minimal time, while the computational cost of the negotiation process, excluding the required by the negotiation exchange, is negligible. Total execution time for these, admittedly small-scale, scenarios, on contemporary hardware, takes only seconds. For larger scenarios, and more ambiguous domains, the time required for plan recognition and planning is expected to increase, although ELEXIR and PKS, as well as the negotiation process, scale well. Experience thus far indicates that both scenarios and domains can substantially increase in size before computational costs, and thus execution time, become an issue.

**Scenario 1:** We begin with the base-case scenario. In this scenario, the plan recognizer correctly identifies the initiator's goal of setting the table, as well as the subgoals the initiator would like the supporter to fulfil. The hypothesis the negotiator examines first is given by:

$$(0.8, [\{\mathit{SetTable} : \{\mathit{SetSpoons}, \mathit{SetPlates}, \mathit{SetGlasses}\}\}]).$$

In this scenario, there is no need for a directed search of the hypothesis structure supplied by ELEXIR. Using this hypothesis, the negotiation then takes the form in Figure 4.

Once completed, the *SetSpoons*, *SetPlates*, and *SetGlasses* subgoals are syntactically translated into PKS goals and the planner attempts to generate a plan. For example, the partial plan for the *SetPlates* subgoal may be:

```
grasp(left,sidetable,plate1),
grasp(right,sidetable,plate2),
putdown(left,table_pos1,plate1),
putdown(right,table_pos2,plate2).
```

(The plans for the other two subgoals will be similar.) Since the hypothesis and the resulting plan(s) are both known beforehand, they can be used to verify that the experimental results match the expected outcome in this scenario.

**Scenario 2:** This scenario extends the first scenario, and is designed to test the use of directed search to correctly identify the initiator's goal from the hypothesis structure supplied by ELEXIR. In particular, the search focuses on high-level goal identification during negotiation, with the initiator rejecting the hypothesis initially presented by the supporter.

In the first iteration of the negotiation process, the supporter presents the initiator with the following hypothesis:

$$(0.8, [\{\mathit{CleanForks} : \{\mathit{WashForks}\}\{\mathit{PutAwayForks}\}\}]).$$

This hypothesis incorrectly identifies the initiator's goal to be that of cleaning the forks. The initiator rejects this hypothesis, with the supporter moving to the next most probable hypothesis, thus iteratively negotiating with the initiator until the correct goal is found. The number of negotiation iterations can be reduced by adding further reasoning logic about the hypothesis. For simplicity, the next hypothesis correctly identifies the goal of the initiator, so further directed search and negotiation iterations are unnecessary. The correct hypothesis is then the same as the one in Scenario 1:

$$(0.8, [\{\mathit{SetTable} : \{\mathit{SetSpoons}, \mathit{SetPlates}, \mathit{SetGlasses}\}\}]).$$

Negotiation would then take the form as shown in Figure 5.

The remainder of the process then follows the one given in the first scenario: the subgoals are translated for use by PKS; the planner builds plans for setting the plates, spoons, and glasses; and the supporter performs the plan.

This scenario demonstrates that by considering all hypotheses, the framework can recover from an initially incorrect identification of the initiator's goal through the use of a lightweight negotiation strategy and directed search of the hypothesis structure provided by ELEXIR.

**Scenario 3:** The final scenario we consider is designed to test the framework when dealing with the situation in which the goal of the plan pursued by the initiator is correctly identified, but one (or more) of the hypothesised subgoals is not, and is thus rejected by the initiator. If this happens, the supporter, using directed search of the hypothesis structure, will

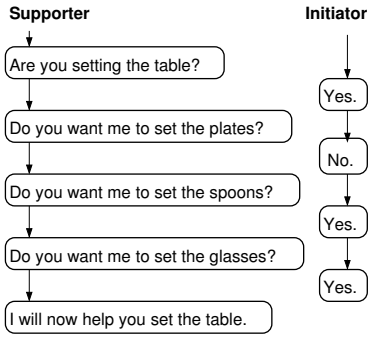


Figure 6: Negotiation in Scenario 3.

iteratively negotiate with the initiator until it finds an acceptable subgoal. It is possible for the supporter to run out of subgoals, if none of the (remaining) subgoals contained in the hypothesis are acceptable to the initiator. If this occurs, the supporter can then revert back to the hypothesis structure to find another hypothesis with the same goal, and continue negotiation with the initiator to see if the (other) subgoals are acceptable. This eventuality is not examined in this scenario due to lack of space. Instead, this scenario considers the same hypothesis as in the first scenario:

$$(0.8, [\{SetTable : \{SetSpoons, SetPlates, SetGlasses\}\}])$$

with negotiation taking the form shown in Figure 6.

The remainder of this process differs from the above scenarios in that the rejected subgoal is not translated and passed to the planner. Instead, only a plan for setting the spoons and glasses is built and performed by the supporter.

This scenario demonstrates that by using ELEXIR's hypothesis structure, the initiator is not limited to accepting all subgoals in a hypothesis: the framework provides enough flexibility for the initiator to decide how, and in which way, he wants to be helped, without the need for elaborate reasoning or goal decomposition on the part of the supporter.

## Discussion

The three experimental scenarios demonstrate that our approach successfully generates cooperative plans: for each scenario, ELEXIR interprets the observations correctly, supplying the correct hypothesis structure to the negotiation process; and the negotiator subsequently presents PKS with the expected subgoals, with the planner producing the correct plans. Thus, in each case the framework produces the expected behaviour, thereby validating the process.

However, the framework also relied on certain assumptions concerning the knowledge of the initiator and supporter. For instance, the plan inferred by the supporter is never shared with the initiator, and this approach does not generate plans with joint actions, where multiple agents must coordinate to perform the same task (e.g., lifting a table). Instead, it only generates independent action sequences for the supporter once there is mutual agreement as to the supporter's subgoals. It is also possible that different agents might use different terms to refer to the same objects. If there is sufficient disagreement on such terms, negotiation will simply break down in the face of failed communication. Likewise, a high degree of overlap between the knowledge of the agents, and a tighter correspondence in the names used to identify domain concepts, should give rise to situations where cooperation is more easily negotiated.

In this work, we have also focused on the importance of the supporter being *proactive* in suggesting goals that it could help the initiator with, based on an understanding of the initiator's plan as identified through plan recognition. While an alternative strategy on the part of the supporter may be to simply ask the initiator how it can help, this is not the focus of our approach. For instance, if the initiator is a human, and the supporter is an artificial agent, the human may be forced to respond to a large number of requests (including clarifications) as to what the initiator is doing. Conversely, the approach in the paper could be adapted to scenarios where an initiator may tell a supporter to achieve certain subgoals. In this case, we could simply bypass the plan recognition process and negotiation stages, using goal translation to pass goals directly to the planner. However, both scenarios require knowledge of the terms used by the initiator, which could be much more extensive than the restricted domain descriptions we work with.

During plan execution, there is no direct reasoning of goal changes on the part of the initiator, except as detected through additional plan recognition. Similarly, the adoption by the initiator of a subgoal assigned to the supporter may result in the initiator performing tasks that have already been planned by the supporter. In such a case, we rely on plan execution monitoring and replanning techniques to generate appropriate behaviour to avoid a duplication of tasks.

Another representational problem that must be overcome involves the correspondence between the domain descriptions used by the plan recognizer and the planner. In particular, it is not unusual for a plan recognizer and a planner to have different representations for the same domain, resulting from differences in the underlying representation languages and problems being solved. However, since the plan recognizer and planner must operate within the same reasoning framework, the onus is currently placed on the domain designer to ensure that domains are appropriately engineered to interoperate correctly. One area of future work is to find a common representation that can be used for both tasks, or to automatically induce one representation from the other.

Finally, in this first stage of our work we have placed greater emphasis on the role of plan recognition, compared with that of planning. However, a key direction of future work is to extend our approach to more complex real-world domains, such as those involving incomplete information and uncertainty, where we can take advantage of PKS's ability to build plans with sensing actions (including communicative actions (Petrick and Foster 2013)) to gather information from the world or other agents at execution time. For instance, if in the example scenario the supporter agreed to place wine glasses on the table, then it may first need to query the initiator as to who is drinking wine (and what type of wine) to ensure the table is properly set. One way to do this is by building a contingent plan with information-

gathering actions completely at the planning stage. Thus, plans could be significantly more complex compared to those in the experimental scenarios.

Another potential use of the planner in the next phase of the work is to address the problem of subgoal achievability during the negotiation stage. Currently, the supporter proposes subgoals to the initiator without determining a priori whether those goals are actually achievable by the supporter. Instead, we are exploring the feasibility of trying to generate (partial) plans for particular subgoals at negotiation, in an attempt to limit the supporter's subgoal proposals to achievable subgoals (or subgoals that at least appear likely to be achievable). While it is not expected that this can be done for all subgoals, due to the time that plan generation could take in complex domains, we are nevertheless exploring this approach as a possibility in smaller domains. One additional advantage of such a technique is that in cases where the supporter knows a subgoal is achievable, the supporter could also explain *how* the subgoal could be achieved, by presenting or summarising the plan. Such an approach may also lead to further opportunities for collaborative behaviour between the supporter and initiator as part of such plans.

## Conclusion

This paper presented a framework for combining plan recognition and automated planning to produce collaborative behaviour between a pair of agents. Successful integration of the plan recognition and planning components centred around appropriate subgoal identification by the plan recognizer, combined with a lightweight negotiation process which generated goals to be used by the planner for constructing appropriate action sequences. A set of experiments demonstrated the potential of our approach, and helped motivate our ongoing and future work to extend these techniques to more complex real-world situations.

## Acknowledgements

The research leading to these results has received funding from the European Union's Seventh Framework Programme under grant no. 270273 (XPERIENCE, `xperience.org`) and grant no. 610917 (STAMINA, `stamina-robot.eu`).

## References

Augusto, J. C. 2007. *Intelligent Computing Everywhere*. London: Springer. chapter Ambient Intelligence: The Confluence of Ubiquitous/Pervasive Computing and Artificial Intelligence, 213–234.

Bauer, A. M.; Wollherr, D.; and Buss, M. 2008. Human-robot collaboration: a survey. *International Journal of Humanoid Robotics* 5(1):47–66.

Blaylock, N., and Allen, J. 2003. Corpus-based statistical goal recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1303–1308.

Boutilier, C., and Brafman, R. 2001. Partial-order planning with concurrent interacting actions. *Journal of Artificial Intelligence Research* 14:105–136.

Brafman, R., and Domshlak, C. 2008. From one to many: Planning for loosely coupled multi-agent systems. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 28–35.

Brenner, M. 2003. A Multiagent Planning Language. In *Proceedings of the Workshop on PDDL at ICAPS 2003*.

Chandrasekaran, B., and Conrad, J. M. 2015. Human-robot collaboration: A survey. In *Proceedigs of the IEEE SouthEastCon*, 1–8.

Crosby, M.; Jonsson, A.; and Rovatsos, M. 2014. A single-agent approach to multiagent planning. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 237–242.

Fikes, R. E., and Nilsson, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2:189–208.

Fong, T.; Thorpe, C.; and Baur, C. 2003. Collaboration, dialogue, and human-robot interaction. In *Robotics Research, Volume 6 of Springer Tracts in Advanced Robotics*. Springer. 255–266.

Foster, M. E.; Giuliani, M.; Müller, T.; Rickert, M.; Knoll, A.; Erlhagen, W.; Bicho, E.; Hipólito, N.; and Louro, L. 2008. Combining goal inference and natural-language dialogue for human-robot joint action. In *ECAI Workshop on Combinations of Intelligent Methods and Applications*.

Geib, C. W. 2009. Delaying commitment in probabilistic plan recognition using combinatory categorial grammars. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1702–1707.

Giuliani, M.; Foster, M. E.; Isard, A.; Matheson, C.; Oberlander, J.; and Knoll, A. 2010. Situated reference in a hybrid human-robot interaction system. In *Proceedings of the International Natural Language Generation Conference (INLG)*, 67–75.

Hawes, N.; Sloman, A.; Wyatt, J.; Zillich, M.; Jacobsson, H.; Kruijff, G.-J. M.; Brenner, M.; Berginc, G.; and Skočaj, D. 2007. Towards an integrated robot with multiple cognitive functions. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 1548–1553.

Henning, M. 2004. A new approach to object-oriented middleware. *IEEE Internet Computing* 8(1):66–75.

Hoogs, A., and Perera, A. A. 2008. Video activity recognition in the real world. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 1551–1554.

Karpas, E.; Levine, S. J.; Yu, P.; and Williams, B. C. 2015. Robust execution of plans for human-robot teams. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 342–346.

Kautz, H. A. 1991. A formal theory of plan recognition and its implementation. In Allen, J. F.; Kautz, H. A.; Pelavin, R. N.; and Tenenberg, J. D., eds., *Reasoning About Plans*. Morgan Kaufmann. 69–126.

Kennedy, W. G.; Bugajska, M. D.; Marge, M.; Adams, W.; Fransen, B. R.; Perzanowski, D.; Schultz, A. C.; and Trafton, J. G. 2007. Spatial representation and reasoning for human-

robot collaboration. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 1554–1559.

Liao, L.; Fox, D.; and Kautz, H. A. 2005. Location-based activity recognition using relational Markov networks. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 773–778.

Lochbaum, K.; Grosz, B.; and Sidner, C. 1990. Models of plans to support communication: An initial report. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 485–490.

Modi, P. J.; Shen, W.-M.; Tambe, M.; and Yokoo, M. 2003. An asynchronous complete method for distributed constraint optimization. In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 161–176.

Nau, D. S. 2007. Current trends in automated planning. *AI Magazine* 28(4):43–58.

Newell, A. 1982. The Knowledge Level. *Artificial Intelligence* 18(1):87–127.

Nwana, H. S. 1996. Software agents: an overview. *The Knowledge Engineering Review* 11(3):205–244.

Pandey, A. K.; Ali, M.; and Alami, R. 2013. Towards a task-aware proactive sociable robot based on multi-state perspective-taking. *International Journal of Social Robotics* 5(2):215–236.

Petrick, R., and Bacchus, F. 2002. A knowledge-based approach to planning with incomplete information and sensing. In *Proceedings of the International Conference on Artificial Intelligence Planning and Scheduling (AIPS)*, 212–221.

Petrick, R., and Bacchus, F. 2004. Extending the knowledge-based approach to planning with incomplete information and sensing. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2–11.

Petrick, R., and Foster, M. E. 2013. Planning for social interaction in a robot bartender domain. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 389–397.

Pollack, M. 1990. Plans as complex mental attitudes. In *Intentions in Communication*. MIT Press. 77–103.

Schrempf, O. C.; Hanebeck, U. D.; Schmid, A. J.; and Worn, H. 2005. A novel approach to proactive human-robot cooperation. In *Proceedings of the IEEE International Symposium on Robot and Human-Robot Interactive Communication (RO-MAN)*, 555–560.

Steedman, M. 2000. *The Syntactic Process*. MIT Press.

Talamadupula, K.; Briggs, G.; Chakraborti, T.; Scheutz, M.; and Kambhampati, S. 2014. Coordination in human-robot teams using mental modeling and plan recognition. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2957–2962.

Zender, H.; Jensfelt, P.; Óscar Martínez Mozos; Kruijff, G.-J. M.; and Burgard, W. 2007. An integrated robotic system for spatial understanding and situated interaction in indoor environments. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 1584–1589.